Contents lists available at SciVerse ScienceDirect

CIRP Annals Manufacturing Technology

Journal homepage: www.elsevier.com/locate/cirp

Evolutionary algorithms for generation and optimization of tool paths

Aydin Nassehi (2), Wesley Essink, Jack Barclay

Department of Mechanical Engineering, University of Bath, Bath, BA2 7AY, United Kingdom

The efficiency of many manufacturing processes is dependent on the properties of the motion path that an end-effector or cutting tool follows. Consequently, algorithms have been developed for generating such tool paths with the aim of optimizing various objectives. In this paper, evolutionary algorithms for generation of continuous motion paths have been investigated to propose a novel method for creation and adaptable optimisation of paths that allows the objective to be modified without revising the algorithm. An example test case based on milling with a sequence of objectives has been used to demonstrate the advantages of the proposed method.

Tool Path, Genetic, Optimization

1. Introduction

Many manufacturing processes are heavily affected by the selection of an appropriate motion path for an end effector: machining, laser welding, additive manufacturing methods relying on material deposition or selective sintering are examples of such processes. The current practice is to use specific algorithms for creating efficient tool paths that optimize a certain objective. Examples can be readily found in the literature for five axis milling [1], wire and arc additive manufacturing [2] and turning [3] amongst others. The motion paths can either be point-to-point as is the case for drilling machines and spot welding robots or continuous as is the case for milling and turning where the motion path describes the continuous movement of the tool relative to the workpiece and is hence termed tool path. In this paper, a novel and flexible methodology based on evolutionary computing for generation and subsequently optimization of continuous tool paths is proposed and presented. Although, much of the logic presented can be applied to all continuous motion path problems, the paper specifically focuses on milling tool path generation as the practical example. The particular focus is on roughing tool paths for 3-axis milling as the effects of the different algorithms for tool path generation would be more prominent and observable due to the high volume of material removal.

In the next section, an overview of current approaches for generation of tool paths is presented to identify the research gap that would be addressed by the development of an objective based tool path generation method. This is followed by a description of the conceptual model that is used to formulate the optimization problem and transform the decision variables to a format suitable for addressing computationally. An algorithm for solving the resulting problem is then presented to construct a prototype demonstrator for assessing the flexibility of the approach through the use of a well known industrially inspired test case in conjunction with a series of different objectives.

2. Tool path generation in milling

Tool path generation algorithms for milling have historically been designed as solving a purely geometric problem—i.e. to generate a path that fully covers an area. Simple 2D clearing methods such as zigzag, contour-parallel, or spiral tool paths, are easily described algorithmically with a few parameters that can be changed to affect the outcome. Many studies have been performed that look into the optimization of these machining parameters to improve cycle times, reduce costs, improve accuracy, as well as a host of other objectives [4,5,6,7]. Many of these studies use evolutionary computation techniques to perform the optimization, leading to an improved or optimal set of feed rates, cutting speeds, depths of cut and more. The problem of initial tool path generation however, has not been widely considered as an optimization problem. The existing research in the area is mainly focused on sequencing of a number of discrete elements into an overall plan. Examples of technologies considered within this perspective include hole drilling [8], CNC laser cutting [9], robot laser welding [10], and non-productive machining time during milling [11]. In these scholarly works, the sequencing problem is modeled as a Travelling Salesperson (TSP) type problem with the aim to minimize the movement between holes, welds, or cuts. As for continuous tool path generation, Oysu and Bingul [12] offered an approach for milling tool path generation based on optimization of sequences of predetermined contours and generated good results where there was a clear logic for choosing the contours *a priori*. In this paper, generation of continuous tool paths is modelled as a general optimization problem with a strong emphasis on flexibility so that the objective of the optimization can be changed without necessitating changes in the algorithms.

3. An optimization framework for path generation

Optimization models can be discrete or continuous; for optimization of machining parameters such as step over or feed rate, a continuous optimization method may be most suitable due to the smoothness of the objective function—i.e. the value of objective function at a point can be used to infer about the value of the function at nearby points.

For tool path generation, however, the combinatorial nature of considering potential paths lends itself better to optimizing using a discretized model. Discretizing the problem will provide a structure to the solutions and allow the use of well-established computationally efficient optimization methods.

To generate a simplified discrete model of the milling process, a grid of points is superimposed over the geometry of the part being machined. A solution to the problem is then defined as an ordering of points to be visited on this grid; a complete tool path is produced if the cutting tool tip visits all points on the grid. Makhanov et al. [13], present a variable grid generation methodology that adapts to various areas milling to minimise machining error. For rough machining, however, a simple equidistant grid with spacing smaller than the tool diameter would provide sufficient resolution and reduces computational load. Furthermore, tool paths can be generated on feature-by-feature basis with a feature defined as an area that can be machined without removing the tool from the workpiece; although in order to optimize the path, the tool may be retracted to the safe plane and positioned elsewhere in the feature.

In order to discretize a milling feature the following algorithm is carried out:

1. Identify the feature boundary as a set of interconnected polylines and curves.
2. Offset the polylines and the curves by tool radius to determine the boundaries of the desired tool positions.
3. Superimpose an equidistant grid that is adjusted according to the cutter diameter and the desired cutter diameter engagement.
4. Delete the points that fall outside the boundaries of desired tool positions.

Fig 1 illustrates the algorithm being carried out on a circular pocket with a square boss. In the resulting set of points, if the tool is positioned at each individual point in a given sequence without crossing the boundary, the part is machined and thus such a sequence of points constitutes a feasible rough milling tool path.

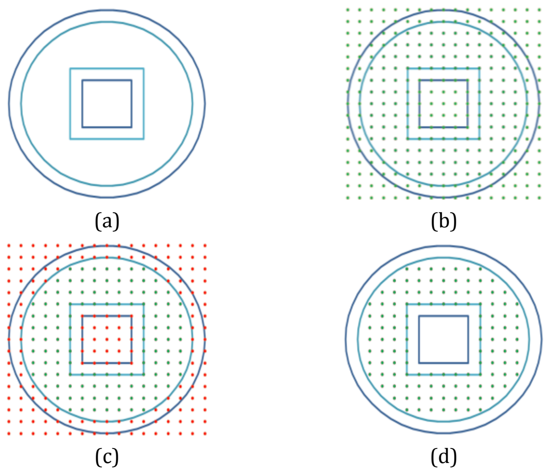

Figure 1. Discretization algorithm: (a) the feature boundary is offset by cutting tool radius; (b) an equidistant grid is superimposed on the feature; (c) points outside the machining boundaries are selected (as shown in red) and (d) removed

4. Formulating the optimization problem

Assume n points are left on the grid. The set $\{1 \dots n\}$ would represent the indices of the points that need to be visited by the cutting tool for the milling operation to succeed. In addition, point 0 is defined as the tool origin from which machining starts and at which the machining process ends. The tool path would be feasible if the tool does not cross the boundaries. Let d_{ij} show the Euclidean distance that the tool should move at feed rate between points i and j with arbitrarily large values when the move is considered illegal; r_{ij} the distance that the tool can move at rapid rates between points i and j; and, X_{ij} a decision variable with the following definition:

$$X_{ij} = \begin{cases} 1 & \text{if point } i \text{ is immediately followed by } j \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Using this notation, the tool path generation constraints can be modelled as follows:

(a) To ensure that all points will be visited at least once:

$$\forall j \in \{0 \dots n\} \sum_{i=0}^{n} X_{ij} > 0 \tag{2}$$

(b) To ensure that the tool leaves each point after visiting for another:

$$\forall j \in \{0 \dots n\} \sum_{i=0}^{n} X_{ij} = \sum_{k=0}^{n} X_{jk} \tag{3}$$

(c) To ensure that every point is followed by a different point and the path moves on:

$$\forall i \in \{0 \dots n\} \;\; X_{ii} = 0 \tag{4}$$

(d) To disallow sub tours so that the entire path would be connected:

$$\forall S \subset \{0 \dots n\} : \;\; S = \emptyset \oplus \sum_{i \in S} \sum_{j \notin S} X_{ij} + X_{ji} \geq 2 \tag{5}$$

(e) For certain objectives the temporal order in which the points are visited is important. It is therefore pertinent to capture the order. Let $T_m = \langle p_1, p_2, \dots, p_m \rangle$ represent the sequence of the points that are visited in a tool path; the length of the sequence m would be equal or greater than the number of points in the grid (n) allowing for multiple visits to a given point. The sequence is defined as follows:

$$\begin{aligned} T_m = \{(i,j): i \in \{1 \dots m\}, j \in \{0 \dots n\}, \\ (k,l) \in T_m \wedge (k,p) \in T_m \Leftrightarrow l = p, \\ \forall x \in \{1 \dots m\} \, \exists y \in \{0 \dots n\}: (x,y) \in T_m\} \end{aligned} \tag{6}$$

Where (i,j) is the ordered pair of integers i and j. Let $T_{m,q}$ indicate the sequence of the first q points visited in T_m where ($q \leq m$), so:

$$T_{m,q} \subset T_m \wedge \forall x \in \{1 \dots q\} \exists y \in \{0 \dots n\}: (x,y) \in T_{m,q} \tag{7}$$

As T_m is essentially a function ($T_m: \mathbb{N} \longrightarrow \mathbb{N}$), $T_m(w)$ would be the wth visited point in the sequence. The sequence can then be linked to the paths chosen using the X_{ij} variables using the following constraint:

$$X_{ij} = 1 \;\Leftrightarrow \exists x \in \{1 \dots m-1\}: T_m(x) = i \wedge T_m(x+1) = j \tag{8}$$

Any sequence T_m and the corresponding selection of X_{ij}s that meet the above constraints would thus constitute a feasible tool path that may be used to produce the part.

5. Objective functions for optimization of tool paths

A series of consecutive objectives have been selected to demonstrate the versatility of the modelling approach and provide a concrete example for the optimization framework. These functions have been selected to represent the typical goals pursued in generation of milling tool paths [14].

5.1. Optimization of cutting time

In order to minimize the cutting time, the values F_c and F_r are assumed to represent the metal cutting feed rate and the rate of movement at rapid speeds in the machine. The objective of the optimization problem can thus be written as :

$$Min \; \sum_{i=0}^{n} \sum_{j=0}^{n} \left(\frac{d_{ij}}{F_c} + \frac{r_{ij}}{F_r} \right) X_{ij} \tag{9}$$

The optimization problem resulting from this objective subject to constraints introduced in section 4 is a linear binary programming problem. This objective converts the distances to times using appropriate feed rates and generates a set of optimal tool paths. A single tool path is then selected as optimal if each point is visited once, effectively converting the optimization problem to a travelling salesperson problem. As multiple visits are allowed, if a point is visited more than once, multiple tool paths are selected with decision points at crossings but all paths will have the same value for the objective function and thus the same machining time. Changing the machine tool used to machine the part, with different rapid and feed rates, could change the optimal tool path; this is in contrast to geometric tool path generation and could have implications for high speed machining (HSM).

5.2. Minimization of jerk

In order to minimize the changes in acceleration, tool paths that keep the tool moving in the same direction for the longest possible period of time should be favoured. As such, the objective function can be defined as follows:

$$Min\ \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=0}^{n} X_{ij}X_{jk}\left(d_{ij}+r_{ij}+d_{jk}+r_{jk}-d_{ik}-r_{ik}\right)^2 \quad (10)$$

This objective function minimizes the turns and consequently leads to lower jerk in the machining process. Excessive jerk can lead to increased wear on the actuators, induce vibrations in the machine structure, and be hard for a controller to accurately follow [15]. The resulting optimization problem is a quadratic binary programming problem.

5.3. Keeping constant cutter engagement

Previous research has shown that keeping constant cutter engagement enables the feed rates to be increased and has a good effect on tool life [16]. Optimizing cutter engagement requires the temporal order of the points that are visited to be known and thus can be written using the previously defined sequence T_m as follows:

$$Min\ \sum_{q=1}^{m-1}\left(d_{T_m(q)T_m(q+1)}\xi\left(\gamma\left(T_{m,q}\right),T_m(q),T_m(q+1)\right)\right)^2 \qquad (11)$$

Where $\gamma(T_{m,q})$ is an iterative function that calculates the geometry of the part after the first q steps of the toolpath T_m have been executed and $\xi(G,i,j)$ a function that calculates the cutter engagement when the tool tip is moved from point i to point j on a part with geometry G. The resulting optimization problem is quadratic.

6. Framework implementation using genetic algorithms

The optimization problem formed in section 5, even in its simplest form, constitutes an NP-hard combinatorial problem that would be computationally difficult to solve using exact methods. With the assumption that every point on the grid will only be visited once, an approach using genetic algorithms (GA), is selected to solve the problem to near optimality. The computational steps for solving the problem are thus defined as follows:

Step 1. Calculate the cutting and rapid distance between grid points: In this step the distances between every pair of points remaining in the grid are calculated. The pairs are then sorted in an ascending order using the cutting distance as a heuristic for a good cutting strategy.

Step 2. Construct an initial population of feasible tool paths: A random point is selected to start the tool path and then the subsequent point that will be visited by the tool is selected from the remaining points. The subsequent point is selected randomly from the remaining points but a higher probability of selection will be given to the points whose link with the first point are ranked higher (i.e. lower distance) in the table generated in step 1. This process continues until all points in the grid are selected. The process is then repeated with other random starting points and random selections to form the initial population of paths for starting the genetic algorithm.

Step 3. Calculate the fitness function for each path: The fitness function (the inverse of the value of the minimization objective function) is then calculated for each individual in the initial population.

Step 4. Selection of paths for reproduction: Tool paths are then selected using fitness proportionate selection method.

Step 5. Application of crossover function to generate offspring: As the genes are sequences, a crossover method that would not generate duplicate chromosomes is more likely to yield efficient results and as such the partially matched crossover (PMX) method has been selected. The new generation is thus comprised completely of offspring that number the same as the parent population.

Step 6. Application of mutations: The selected tool paths are then randomly mutated. The mutation operator uses the nearest neighbours approach [17].

Step 7. Recalculation of fitness function: For each of the resulting offspring tool paths, the fitness function is recalculated. The value for the best path is compared to the historic figures from previous generations. The process halts if an improvement has not been seen for a number of generations, with the tool path generating the best fitness function selected as the near optimal solution; otherwise the process is repeated from step 4.

7. Test case: an industrially inspired machining part

In order to test the algorithm presented in section 6, a prototype implementation has been created using the Java programming language. The prototype gets the geometry data from a STEP-NC [18] formatted process plan, generates the grid as presented in section 3 and performs the genetic algorithm presented in section 6. The parameters chosen for the genetic algorithm are: number of individuals in a population = 25; mutation is applied to all offspring and the maximum crossover length is 0.5 of a chromosome. The algorithm has been successfully tested on a selection of prismatic milling parts and the attributes of the tool paths generated by the prototype are presented based on the industrially inspired "fishhead" part [18] as an exemplar (shown in Fig 2).

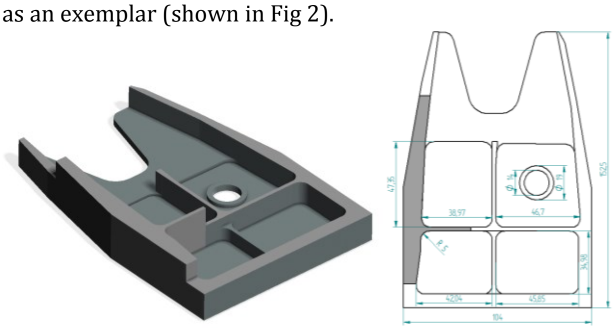

Figure 2. The "Fishhead" part, standard part based on components in the aerospace industry and the selected machining plane

Three tool paths have been generated by running the prototype with the objective functions presented in section 5 and the results have been analysed for the selected machining plane. The generated paths are henceforth referred to as the time oriented tool path (TOT) based on objective 5.1, straightness oriented tool path (SOT) based on objective 5.2 and consistent engagement tool path (CET) based on objective 5.3. Furthermore a path based on an arbitrary linear combination of all three objectives is studied (MUL) to explore the possibilities of multi objective optimisation in the future. Figure 3 shows visualizations of the tool paths generated on the test plane.

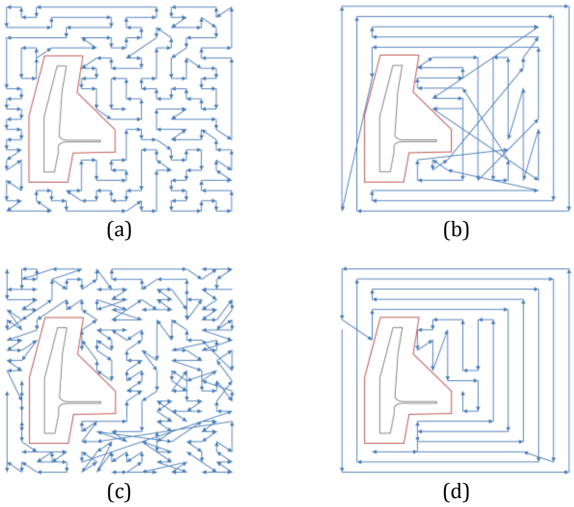

Figure 3. The generated tool paths: (a) TOT (b) SOT (c) CET (d) MUL

Table 1 shows a comparison of basic properties for the generated tool paths. Whilst, as expected, TOT has produced the shortest path and SOT the least amount of turns, MUL has managed to produce a tool path with a good balance in these properties. An attempt was made to use the branch-and-cut method to calculate the optimum value for TOT. Whilst the method failed to produce results within 12 hours on a quad-core Intel Core i7 3.5Ghz CPU, a lower bound of 2860mm was established by calculating the optimum for the linear relaxation of the problem indicating that the results achieved using TOT is within 3% of the optimum in the worst case.

Table 1 Comparison of basic properties for the generated paths for the example feature

Property	TOT	SOT	CET	MUL
Total length (mm)	2937	3615	3999	3012
Number of turns	185	41	260	42
Convergence time	~10s	~10s	~30s	~30s

Figure 4 shows histograms of cutter engagement (measured as the engaged percentage of cutter diameter). It can be seen that CET generates the most consistent engagement with MUL offering a good balance.

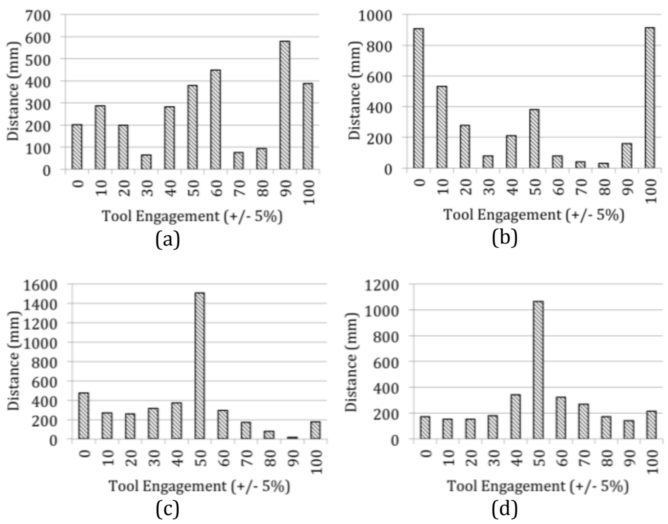

Figure 4. Distance covered by the cutting tool at different levels of cutter engagement for (a) TOT (b) SOT (c) CET (d) MUL

8. Conclusions and future work

Current methods for generating tool paths are designed to optimize preselected properties of the path making them inflexible; in order to optimize a different property, usually, a completely different algorithm has to be used by the path generation system.

In this paper, the authors focused on the milling process and presented a discretization framework together with a flexible evolutionary computational approach for generating tool paths for prismatic features that allows various properties to be optimized without changing the algorithm. The algorithm was shown to be effective when optimizing three different properties (time, straightness and cutter engagement) and generated tool paths with large differences in the values of these properties.

The proposed method has the unique advantage that by only modifying the optimization objective, different tool path for meeting various aims can be generated without the necessity of modifying the underlying algorithms.

This flexibility would allow manufacturing processes to be modified according to evolving objectives in the business context. In the future, the aim is to utilize the framework to achieve more complex multi objective optimization of tool paths to further enhance the promising results of simple linear combination of various objective functions.

References

[1] Lauwers, B., Plakhotnik, D., 2012, Five-axis milling tool path generation with dynamic step-over calculation based on integrated material removal simulation, CIRP Annals – Manufacturing Technology, 61/1:139-142.

[2] Ding, D., Pan, Z., Cuiuri, D., Li, H., 2014, A tool-path generation strategy for wire and arc additive manufacturing, The Int. J. of Adv. Manuf. Tech. 73/1-4:173-183.

[3] Zhang, X., Liu, R., Nassehi, A., Newman, S. T., 2011, A STEP-compliant process planning system for CNC turning operations, Robotics and Comp. Int. Manuf., 27/2:349-356.

[4] Zain, A.M., Haron, H., Sharif, S., 2010, Application of GA to optimize cutting conditions for minimizing surface roughness in end milling machining process, Expert Sys. with Applications. 37/6:4650-4659.

[5] Rai, J.K., Brand, D., Slama, M., Xirouchakis, P., 2011, Optimal selection of cutting parameters in multi-tool milling operations using a genetic algorithm, I. J. of Prod. Res. 49/10:3045-3068.

[6] Ganesan, H., Mohankumar, G., Ganesan, K., Ramesh Kumar, K., 2011, Optimization of machining parameters in turning process using genetic algorithm and particle swarm optimization with experimental verification, I. J. of Eng. Sci. and Tech. 3/2:1091-1102.

[7] D'Addona, D.M., Teti, R., 2013, Genetic Algorithm-based Optimization of Cutting Parameters in Turning Processes, Procedia CIRP, 7:323-328.

[8] Huizar, C., Montiel-Ross, O., Sepúlveda, R., Delgadillo, F.J.D., 2013, Path planning using clonal selection algorithm, Recent advances on hybrid intelligent systems – Studies in Computational Intelligence, 451:303-312.

[9] Vaupotic, B., Kovacic, M., Ficko, M., Balic, J., 2006, Concept of automatic programming of NC machine for metal plate cutting by genetic algorithm method, J. of Achievements in Mat. and Manuf. Eng., 14/1-2:131-139.

[10] Erdős, G., Kemény, Z., Kovács, A., & Váncza, J., 2013, Planning of remote laser welding processes, Procedia CIRP, 7:222-227.

[11] Gupta, A. K., Chandna, P., Tandon, P., 2011, Hybrid genetic algorithm for minimizing non productive machining time during 2.5 D milling, I. J. of Eng., Sci. and Tech., 3/1:183-190.

[12] Oysu, C, Bingul, Z., 2009, Application of heuristic and hybrid-GASA algorithms to tool-path optimization problem for minimizing airtime during machining, Eng. App. of Artificial Intelligence, 22/3:389-296.

[13] Makhanov, S. S., Batanov, D., Bohez, E., Sonthipaumpoon, K., Anotaipaiboon, W., Tabucanon, M., 2002, On the tool-path optimization of a milling robot, Comp. & Ind. Eng., 43/3:455-472.

[14] Altintas, Y., Kersting, P., Biermann, D., Budak, E., Denkena, B., Lazoglu, I., 2014, Virtual process systems for part machining operations, CIRP Annals-Manufacturing Technology, 63/2:585-605.

[15] Gasparetto, A., Zanotto, V., 2007, A new method for smooth trajectory planning of robot manipulators. Mechanism and Machine Theory, 42/4:455-471.

[16] Schulz, H., Hock, S., 1995, High-speed milling of dies and moulds—cutting conditions and technology, CIRP Annals-Manufacturing Technology, 44/1:35-38.

[17] Essink, W. P., Nassehi, A., Newman, S. T., 2014, Toolpath generation for CNC milled parts using genetic algorithms, Enabling Manufacturing Competitiveness and Economic Sustainability, Springer, 189-193.

[18] Hamilton, K., Hascoet, J.-Y., Rauch, M., 2014, Implementing STEP-NC: exploring possibilities for the future of advanced manufacturing, Modern Mechanical Engineering, Materials Forming, Machining and Tribology, Springer, 199-239.